V&V of ISHM for Space Exploration

Lawrence Markosian
QSS Group, Inc./NASA Ames Research Center

Martin S. Feather and David Brinza
Jet Propulsion Laboratory,
California Institute of Technology

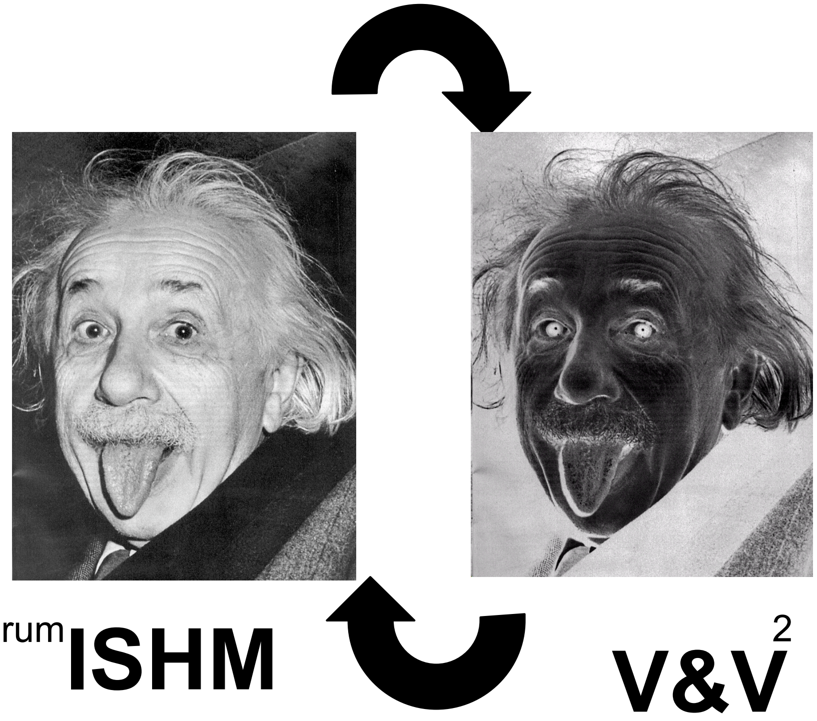
Fernando Figueroa
NASA Stennis Space Center

ISHEM Forum 2005

Two Purposes of this Talk

Our chapter in…

Connect two complementary communities of experts



What will **inhibit** use of ISHEM in Human Space Exploration?

IT CAN'T MEET THE NASA HUMAN RATING REQUIREMENT…

IT CAN'T BE CERTIFIED!

How NASA views ISHM

Class A Human Rated Software Systems

"Applies to all space flight software subsystems (ground and flight) developed and/or operated by or for NASA to …

Examples of Class A software for human rated space flight include … failure detection, isolation and recovery …"

ISHM system is Class A software

Mandates an approach to V&V and certification that has close parallels with those followed in other safety-critical application areas

E.g., FAA Software Verification

The DO-178B/ED-12B Software Verification Process defines specific verification objectives that must be satisfied; these include:

a. Verification of software development processes

b. Review of software development life cycle data

c. Functional Verification of software
- **i. Requirements-based testing and analysis**
- **ii. Robustness testing**

d. Struc al e i ll ughly e & e code

Good process

Scrutinize

uirements, designs, …

reviews, inspections, …

Two problems with this:

Infeasible – colossally expensive to do!

Insufficient – even if could, it wouldn’t be enough

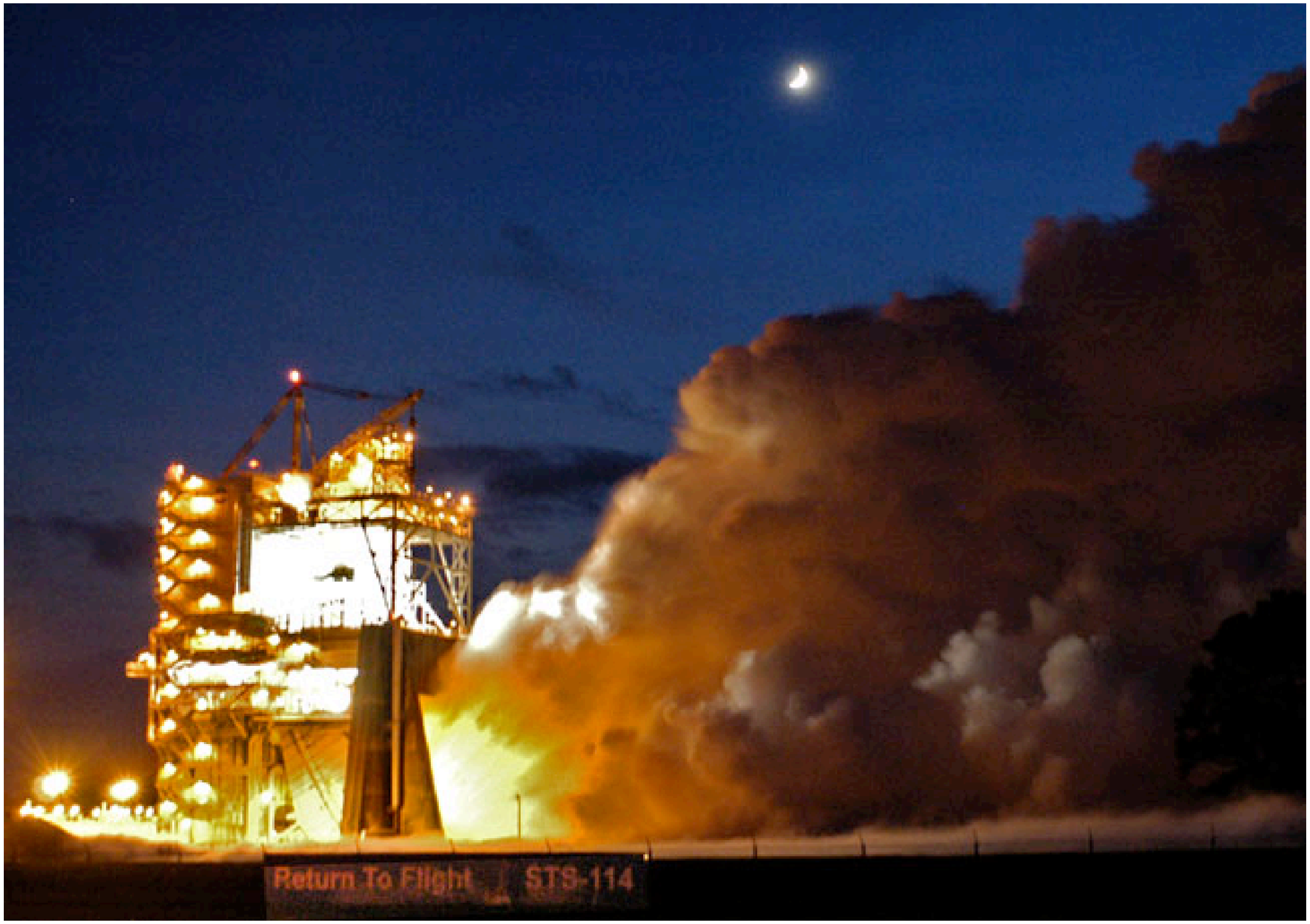
…Lessons Learned…

[Binkley, Cheng, Smith & Tosney]

“Test as you fly – fly as you test”

But… tests can be expensive…

Another Hot-Fire Test of SSME at SSC

- 

And *another*, and *another*…

- **Thirty** hot-fire tests of AHMS for SSME
- These are mainly for showing that there are no *false positives*
- Many additional tests in the hardware-in-the-loop testbed to show small number of *false negatives*.

Traditional software… Fault Detection, Isolation & Recovery …ISHM

Traditional

Modest number of nominal behaviors,
with small variants / parameter ranges

FDIR

Modest number of off-nominal behaviors
Avert catastrophe-e.g., revert to safe
mode (Paula Morgan – Cassini)

e.g., "Limit-sensing software … straightforward … easy to certify" [McCann & Spirkovska]

ISHM

Huge number of off-nominal behaviors
Avert catastrophe & **maintain capability**
(**huge number of reactions**)

A design requirement

Space systems shall be designed so that no two failures result in crew or passenger fatality or permanent disability.

A testing requirement

The Program Manager shall perform testing to verify and validate the performance, security, and reliability of all critical software across the entire performance envelope (or flight envelope) including mission functions, modes, and transitions

B. Hughitt: *100% verification of critical product attributes*

Uh-oh: if 200 possible failures, 40,000 *pairs* of possible failures

Don’t Rely on Waivers!

WAIVER

WAIVER WAIVER

WAIVER WAIVER

WAIVER WAIVER

WAIVER WAIVER VER

R

WAIVER WAIVER

WAIVER WAIVER

Traditional software architectures

Control software + **Data (parameters)**

Novel ISHM software architectures

rule-based expert systems
case-based reasoning systems
model-based reasoning systems
learning systems
probabilistic reasoning systems

[Patterson-Hine,
Aaseng,
Biswas &
Narasimhan]

General-purpose reasoner + **System-specific rules/models**

Emerging V&V techniques, suited to ISHM, may save the day!

General-purpose reasoner + **System-specific rules/models**

Yield an answer *and its rationale*

V&V once, then *reuse!*

Rules/models *themselves* mathematically analyzable

Examples: see our paper

Diagnosability

- One important property of vehicle models for ISHM: they should support *diagnosability* of a class of faults.
  - Detect all the target faults
  - Distinguish between different target faults
- Work by Charles Pecheur (formerly of NASA Ames) et al.
  - Uses “model checking” to explore all possible states of the model

Other applications of model checking

- Verification of planning systems
- V&V of programs
  - For example, SPIN for C code, Java PathFinder for object-oriented code, etc
  - “Out of the box” capability for detecting concurrency pathologies – embedded/reactive systems
  - Can be applied to V&V of other properties of code as well.
  - Test case generation

Verification of Core Algorithms

- For example, voting schemes for fault detection
- V&V technologies include formal methods such as theorem proving (Rushby, SRI; Miner, LaRC)

Verification of procedural code

- Static analysis
  - Often oriented toward “structural” defects
  - More advanced tools can target application-specific program properties
  - See Guillaume Brat’s poster
- Runtime analysis
  - Identify concurrency pathologies by analyzing test data even when the test doesn’t trigger a bug
- Program model checking
  - [Mentioned earlier]

What can YOU do?
Please, please, please…
design with V&V in mind!

- System and software requirements – formulate to be feasible for V&V
- Design –
  Irem Tumer, Andy Hess: "…HM in system trades…" include V&V in these trades
  Serdar Uckun: "…cost & benefit of ISHM…" include its V&V in these evaluations
  Architect to permit modification, change, …

Plan to advance and introduce ISHM incrementally in concert with advancement of V&V

- Ryan Mackey: incrementally introduce ISHM as confidence is gained
- Continue to do things right: none of the V&V technologies we mentioned will overcome poor development practices earlier in the lifecycle
- **Better connect the V&V community with the ISHM community starting <u>now</u>**!